Computers Math. Applic. Vol. 23, No. 2-5, pp. 179–204, 1992
Printed in Great Britain.

0097-4943/92 $5.00 + 0.00

NETL AND SUBSEQUENT PATH-BASED INHERITANCE THEORIES

RICHMOND H. THOMASON
Intelligent Systems Program, University of Pittsburgh
Pittsburgh, PA 15260 U.S.A.

Abstract—Scott Fahlman's NETL project, completed as an MIT dissertation in 1977, arose out of concern with the problem of high-performance knowledge representation in AI systems. Fahlman proposed a system in which network representations would be implemented on a parallel architecture designed to carry out inheritance reasoning efficiently.

David Touretzky's later work, completed in 1984 as a CMU dissertation, exhibits conceptual and computational problems in NETL, and addresses these by developing a theory of inheritance reasoning and associated *parallel marker propagation* algorithms. This theory, in which paths through networks are analogous to proofs in logic, has become an active area of research in knowledge representation. The methods are like those of logic, but the research concentrates on formalisms that are expressively weak compared to familiar logics, but which may allow inference procedures that are tractable.

This paper tries to provide some perspective on the theory, concepts, and current research trends in this field.

1. INHERITANCE

The idea of inheritance arises naturally in almost any situation in which large amounts of information must be stored on a computer for intelligent processing. To store the information efficiently, to maintain coherence under updates, and to present the knowledge base sensibly to users, it is very useful to organize things so that what is more general can be related to what is more particular, allowing information to be stored at the highest level and transmitted down as needed. This flow of information from subsuming to subsumed items is called "inheritance."

The inheritance metaphor suggests family relationships. It is useful to display these relationships in "family tree" diagrams, which if we confine ourselves to one sex are literal trees in the graph-theoretic sense, but which in the general case will be "tangled," and can be represented as directed acyclic graphs (DAGs).

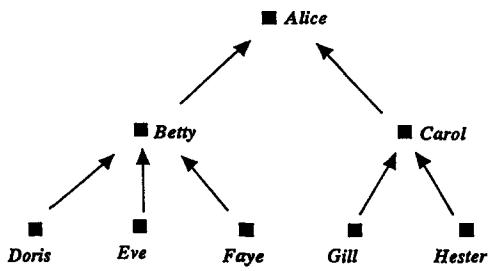

Figure 1. Family tree.

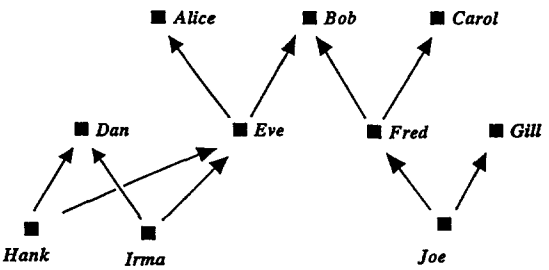

Figure 2. Family DAG.

Despite the lack of standardization in the area, the use of such techniques is ubiquitous in knowledge representation. Surveys such as [1] make this evident. More generally, in 1986 Touretzky listed FRL, KRL, SRL, KL-ONE, SMALLTALK, FLAVORS, LOOPS and ADA among well-known programming and knowledge representation systems incorporating inheritance.[1] Since then, the

The author acknowledges the support of the National Science Foundation under grant IRI-9003165. This paper owes much to comments from David Touretzky and John Horty; their generous help is gratefully acknowledged.
[1]See [2, p. 1] for references.

list has grown—with object-oriented LISP a notable recent addition.[2] Application areas are also expanding. An interesting and rapidly growing special-purpose area is linguistic description; see [4] and [5] for examples of recent applications of nonmonotonic inheritance in the lexicon and other areas of descriptive linguistics.

Scott Fahlman's MIT dissertation, NETL: *A System for Representing and Using Real-World Knowledge*, was completed in 1977 and published as a book in 1979. The book serves as a focus for both previous and later work in network-based knowledge representation and inheritance. Many ideas that preceded Fahlman's work and that influenced it are brought together for the first time in the dissertation, and (especially through Touretzky's work) it has shaped later developments.

Touretzky's 1984 dissertation, subsequently published as [2], concentrates on developing a basis for understanding inheritance; it serves as a basis for the theoretical work that has been done later, by Touretzky and associates in Pittsburgh and by other researchers elsewhere.

The best way to put inheritance theory in perspective, then, and to become acquainted with the leading ideas of the subject, is to begin with Fahlman and Touretzky.

2. NETL

Fahlman aimed high. The work's central position in the research record—as well as the fact that the full scheme has never been fully implemented—are closely related to the ambitious scale of the NETL project. Beginning with the need to usefully manage massive amounts of information in AI applications, Fahlman proposes a system combining network representations with parallel procedures. The idea is to equip a powerful knowledge representation language with a battery of core procedures that—running on a special-purpose NETL machine at least—would perform efficiently even on very large knowledge bases.

Fahlman's self-expressed goals for NETL were:[3]

(1) Compatibility with the parallel network implementation.
(2) [Expressive] Completeness: the system should be able to represent anything that people can.
(3) Semantic precision.
(4) Simplicity and intuitive clarity.
(5) Economical representation.

Knowledge representators have become more cautious since 1977. It has become painfully clear that the most important of these goals—the first three—compete with one another. More recent work on inheritance has not entirely forgotten Fahlman's comprehensive vision, but has tended to concentrate on developing semantic precision, in the hope that this will help at least in the design of smaller-scale inheritance applications, and may eventually furnish insights that could lead to larger and more ambitious systems.

2.1. The NETL *Machine*

Fahlman's motivation for a parallel architecture comes directly from the core AI tradition in knowledge representation, rather than from cognitive psychology or neural modeling. The project was meant to be judged by the performance of implementations and by the solutions it offers to problems of knowledge representation. In the spirit of symbolic AI, these solutions would involve explicitly designed representations and programs rather than learning. And the NETL machine is not homogeneously parallel; in fact it is controlled by a serial machine.

The controlling serial machine communicates with a large array of connected small processors by messages transmitted over a party-line bus, used for sending broadcast commands and for polling the array. A limited number of marker-bits (20 per element in Fahlman's original design) serve to store temporary information; other bits provide an address. The design must somehow enable links of various types between nodes, which can transmit information. That is, the system must provide for local interactions that can change the marker settings of connected nodes.

[2] See [3, Chapter 28].
[3] [6, p. 70–71]

In discussing the feasibility of such a machine, Fahlman speculates without much success [6, Appendix A.2] about possible solutions to the engineering problem of providing a system of this sort with the capability of directly setting arbitrary links. The only feasible solutions seem to involve relaxing literal connectivity, either by grouping nodes into neighborhoods that are sparsely interconnected with each other, or by simulating clusters of elements with networked units that are relatively small considered as serial machines, but much more powerful than the NETL elements. This last idea has proved to be the most promising—though it is obviously expensive to build such a machine—and it inspired the development of The Connection Machine.[4]

Though The Connection Machine has not in fact been extensively used for knowledge representation purposes, and though later theoretical developments have exposed ways in which well motivated inheritance algorithms could exceed the power of a NETL machine, Fahlman's idea of implementing inheritance on a parallel architecture is still powerful and attractive. Though parallel algorithms for inheritance may be incomplete or even unsound, there is reason to hope that they would approximate sound and complete algorithms in most realistic applications, and in fact that many such instances could be automatically identified. Since the NETL architecture already incorporates a serial machine, there is no reason why sound and complete algorithms could not be added to an implementation, and used for appropriate tasks in which their slower performance is acceptable.

These hopes admittedly have not been tested in practice. Nevertheless, much of the work in inheritance theory continues, as Fahlman did, to implement serial simulations of the algorithms, and, as Touretzky did, to relate inheritance problems to mathematical models of a parallel marker passing machine.

Since it is only very recently that a Fahlman-like knowledge base was implemented on a large scale,[5] there is not much information through actual testing about the performance level delivered by such schemes.[6] The practical evaluative dimensions that apply to the NETL system relate (1) to the efficiency and coverage of the related algorithms and (2) to the expressive adequacy of its representation scheme.

2.2. Processing Adequacy

In dealing with the first issue, Fahlman argues informally that procedures such as transitive closure (or shortest-path modifications of transitive closure to take exceptions into account) and intersection will suffice to perform a number of critical AI tasks, including inheritance, conflict checking, and recognition. He then shows that the time complexity of *parallel marker propagation machine algorithms* (PMPM algorithms),[7] when run on the NETL machine, will be linear in the depth of the knowledge base—i.e., linear in the length of the longest chain of IS-A links through the network.[8] Assuming that the taxonomies encountered in representing knowledge are "bushy," or broad but shallow, and that in fact the depth of even very large real-life knowledge bases is bounded by a fairly low constant, this would assure performance in *constant* time for these core algorithms.

2.3. Expressive Adequacy

The second issue—expressive adequacy—is more difficult to address without a full-scale implementation that exercises the system in complex, diverse reasoning tasks. In addressing this issue, Fahlman provides, in appendices to his book, two exercises in representation. (The domains are animal classification and electronic circuits.) In the central chapter of the book, he identifies a number of central representational constructs, and—at least for certain core concepts—shows

[4] See [7].

[5] This is partly an accident. By the time The Connection Machine was developed, the NETL project had been dormant for some time. Few very large knowledge bases have been developed at all; given the cost of simply building one, it is not surprising that researchers have been slow to implement them on a parallel machine.

[6] See [8] and [9].

[7] The term is Touretzky's. See Section 4.5, below, for more information about Touretzky's work on PMPM algorithms.

[8] To be even more precise, the correct measure is the longest shortest-distance IS-A path from one node of the net to another.

how they can be incorporated in the system. This core includes subsumption, negation, roles and relations, and (perhaps) temporal reasoning and the representation and management of contexts. Other constructs that Fahlman discusses, such as definitions involving universal quantification, I do not include in the NETL core, because he does not attempt to provide PMPM algorithms to do the related reasoning, saying only that the reasoning may be approximated cursorily by parallel algorithms and deferred in many cases to the serial processor.

A realistic test of the NETL representation system would almost certainly reveal expressive inadequacies. The system of temporal representation, for instance, is not worked out. And many of the complaints that are urged in [10] about KL-ONE-like systems would also apply to NETL. The system does incorporate central expressive features that should make it generally useful in common-sense oriented domains, ones in which large amounts of common sense knowledge are loosely linked by generalizations that may have exceptions. Fahlman's animal domain is a good example, other examples might include diagnostic knowledge in medicine, and lexical information about a language.

The most important of NETL's features is the idea of "virtual copying." By this Fahlman understands the ability to make a general template, and to apply not only features of this template but relational information to instances. To adapt a well known blocks-world example from [11], the templates of some generic shapes might include the following information:

Slabs:

1. Slabs have a length, which is a number.
2. Slabs have a width, which is a number.
3. Slabs have a depth, which is a number.

Cylinders:

1. Cylinders have a length, which is a number.
2. Cylinders have a width, which is a number.
3. Cylinders have a depth, which is a number.

Arches:

1. Arches have a *lintel*, which is a slab.
2. Arches have a *left column*. An arch's left column is a cylinder which supports its lintel.
3. Arches have a *right column*. An arch's right column is a cylinder which supports its lintel.
4. An arch's left column's length is the same as its right column's length.

When information such as this is inherited to a subsumed concept or object (say a Roman arch), new concepts or objects (a Roman arch's lintel, for instance) have to be hypothesized, and placed in the relations assigned by the template. Fahlman stresses that this inference should not in general actually copy the information from the template. In keeping with the information-management strategy of inheritance systems, "put information at the most general level," we don't, for instance, want to have to create a node representing the Roman arch's lintel unless specific information is known about the lintels of Roman arches. This explains Fahlman's term, "virtual copying."

Fahlman seems right to stress the importance of the virtual copying idea in knowledge representation. It combines a natural, pictorial way of presenting information about types with intuitions about the related reasoning that, in [6] at least, are informal, but that also suggest implementation tactics. The importance of virtual copying (though not under that name) was suggested in Minsky's frame paper, [12], and has been implemented in many of the frame-inspired systems. But the idea also seems to reappear in a variety of domains, from a number of perspectives. The reasoning about roles provided by KL-ONE-style systems is based on very similar ideas. And the idea emerges naturally in unification-based approaches to natural language processing, where one wants to represent information such as the following.

Nouns:

1. Nouns have a number, which is either *singular* or *plural.*

Verbs:

1. Verbs have a number, which is either *singular* or *plural.*

Noun phrases:

1. Noun phrases have a head, which is a noun.

Verb phrases:

1. Verb phrases have a head, which is a verb.

Sentences:

1. Sentences have a subject, which is a Noun phrase.
2. Sentences have a predicate, which is a Verb phrase.
3. A sentence's subject's head's number is the same as its predicate's head's number.

The mechanism of *template inheritance* implemented in PATR-II is based on ideas that again are very similar to the virtual copying notion; see [13, pp. 55–61] for details.

Evidence such as this suggests strongly that Fahlman was right to lay stress on virtual copying as an important concept for knowledge representation.

The NETL system incorporates many other representational ideas that would generally be useful in representation. These include treatments of propositional types, of quantifiers, of events and actions, and of context. Using a link he calls the "existence wire," Fahlman suggests that regions of what might be called "topical space" be associated with nodes; this mechanism affects the way in which roles are managed, and is meant to influence the appropriate restriction of quantifiers and other inferences that are context sensitive.

In general, these ideas are not worked out in [6] as thoroughly as the core "logical" ideas concerning subsumption and relations. Fahlman is frank about lack of precision in some areas; one of the longest sections of the book is titled "Problems." Touretzky feels, in retrospect, that a major problem with the system as Fahlman presented it was lack of attention to the interaction of the many primitives of the system.[9] At the time [6] was written, the conceptual and computational difficulties that could be created by such interactions were still largely unforeseen. The interaction between multiple exceptions and defeasibility, which led to Touretzky's theory of extensions, is one such example; there are many others.

As we will see, subsequent work inspired by NETL has generally aimed at improving semantic precision, and as a result has clarified our understanding of interactions. The price of this trend is to reduce coverage to a much less impressive array of expressive features. We all hope, of course, that this coverage can be increased. The parts of the NETL system that have not been reworked and clarified would all be valuable in practical applications, and certainly address important representation issues that are almost as poorly understood today as they were when [6] was written. For instance, in [14], John McCarthy notes that the lack of a theory of context is a serious shortcoming in the framework he has developed over the years for representing common-sense knowledge. NETL is likely to be a useful source of ideas for a long time to come.

2.4. The NETL *Language*

I have tried to discuss NETL at a notation-independent level that stresses the general ideas. The diagrams and representations of later work on inheritance differ from Fahlman's in many ways, some of them substantive. I will not have much to say about the actual details of NETL, other than a brief illustration in this section.

The following diagram illustrates Fahlman's notation for representing information about VC links and roles.[10]

Nodes are indicated by circles in the diagram and links by lines or arrows. Nodes representing individuals are written as open circles, while nodes representing types are written as filled circles. Various link types are differentiated by styles of arrows or lines. Single shafted arrows with

[9] Personal communication.

[10] The diagram is not copied directly from anything in [6]; I have made some simplifications in the way links are displayed.

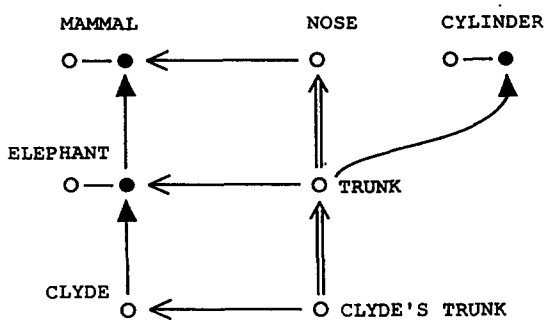

Figure 3. A Fahlman diagram.

closed heads are IS-A or VC links; in general, inheritance diagrams are arranged so that these links point upward. Single shafted arrows with open heads link role fillers to their owners. Double shafted arrows are "map wires," indicating subsumption relations between role fillers. Simple lines connect a type node to its corresponding "set."[11] As you can see from the diagram, sets are treated as individuals.

Though, as I have indicated, there is no commonly accepted diagramming convention for inheritance networks, the diagrams that you see in the work on inheritance are about as similar to one another as different notations in logic. Once the conventions are explained for representing node and link types, they are relatively easy to read.

Besides helping in the theoretical work, these pictures of structured information are one of the most important practical reasons why inheritance systems can be useful in organizing large amounts of information. Inheritance diagrams are more readily grasped, and much easier to work with than more linguistic modes of presentation, such as lists of axioms.

3. TOURETZKY'S MATHEMATICS OF INHERITANCE

David Touretzky's *The Mathematics of Inheritance Systems* was completed as a doctoral dissertation at Carnegie Mellon University in 1984. By that time Fahlman and Touretzky had been working together at Carnegie Mellon University for over five years. Jon Doyle, one of the developers of nonmonotonic logic, was also working at Carnegie Mellon during much of this time. This provided an ideal opportunity to provide a more rigorous foundation for inheritance, an opportunity that Touretzky seized by working closely with both Fahlman and Doyle.

By 1984, work in the general theory of nonmonotonic reasoning was well advanced, and the field was an active area of research.[12]

Touretzky's work provides an intellectual bridge between logical ideas—in particular, theoretical work in nonmonotonic reasoning—and Fahlman-like inheritance networks.[13]

3.1. Methodological Basics

In seeking to provide rigorous foundations for NETL, Touretzky naturally narrows the broad scope of Fahlman's representation language. Touretzky concentrates on just one of Fahlman's AI tasks, the problem of finding the implicit properties of an item in an IS-A hierarchy in which relatively few link types are present. In the central chapters, Touretzky considers only various

[11] Sets in NETL shouldn't be confused with the sets of set theory. The distinction between a type and the corresponding set is used to distinguish what linguists call "distributive plurals" (like 'Elephants are gray') and "collective plurals" (like 'Elephants are scarce'). Since the latter correspond to properties of the type that are not inherited, it is important to mark the distinction somehow if collective information is allowed in a representation language that uses inheritance.

[12] I know of no history of the subject. For an excellent conceptual survey of the field, with references, see Matthew Ginsberg's introduction to [15].

[13] The term 'intellectual bridge' is meant to indicate a two-way path for fruitful transmission of ideas. Formalising the connection has proved to be challenging, and despite some promising results is still an open area of research. More about this later.

sorts of taxonomic links: positive and negative IS-A links, as well as cancellation links.[14] Later chapters develop an account of specific patterns of relational reasoning.

As an appropriate theoretical level at which inheritance can be characterized, and a standard against which the soundness of algorithms can be measured, Touretzky's contribution is unusual—at least, as an approach to nonmonotonic reasoning. Most of the theories in this area, while recognizing that nonmonotonicity is in many ways a radical departure from the logical tradition, have tried to provide a framework that makes nonmonotonic reasoning intelligible in traditional terms, and in in particular, in model–theoretic terms. If Touretzky had followed this trend, he would probably have tried to produce a translation of part of the NETL language into one of the familiar systems of nonmonotonic logic.

Rather than this, he provides a principled account of a number of well chosen examples, which show that inheritance reasoning is more complex than Fahlman had realized. These examples are then used to motivate an *inheritance definition*. The idea of such a definition, in its simplest form, is to create a mathematical model of a network. We can think of the network as containing information, in the form of links. Certain paths composed of these links through the network will correspond to conclusions that follow from the information in the network; to take the simplest possible example, the chain of positive IS-A links in Figure 4 justifies the conclusion that Clyde is gray.

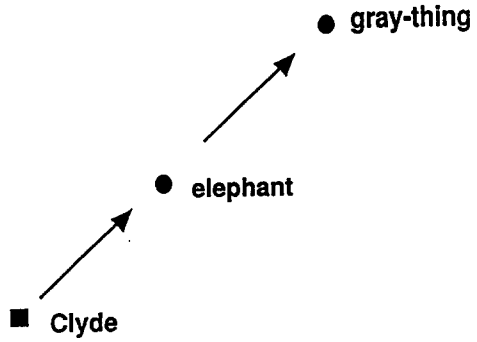

Figure 4. A simple path.

This train of thought suggests that to capture the implicit information contained in a network, we should concentrate on the paths that are permitted by the network. Further, since paths are composed of links, it is natural to provide an inductive definition of the permitted paths. This definition will depend only on a general mathematical model of the network, and so will be implementation independent. The intuitive correctness of the definition can be tested against examples. Also, a rigorous definition will suggest theorems that should be provable about reasoning in networks; proving these theorems provides another test of inheritance definitions. The inheritance definitions that this process yields can be used as standards of correctness for algorithms.[15]

In carrying out this program, Touretzky discovered that by working with diagrams of networks it is possible to develop sophisticated, detailed intuitions about specific instances. It is these intuitions, together with the theory that Touretzky was able to build on them, that make [2] such a fruitful starting place for further research.

3.2. Exceptions, Level Skips and Conflicts

One thing that makes inheritance theory interesting to a logician is the new ways it provides of presenting and organizing patterns of reasoning. This novelty has to do in part with the diagrammatic structure of networks, which can provide information that is lacking in the logical tradition that takes proofs to be arrays of formulas. Features of the reasoning that Fahlman

[14] Where a negative IS-A link between concepts p and q means that p's aren't q's, a cancellation link suspends conclusions as to whether p's are q's.

[15] This account of Touretzky's inheritance theory is simplified in one important respect. The possibility of conflicting paths complicates matters; one response is to make inheritance relative to a set of arbitrary "guesses." This complication is discussed below.

and Touretzky were trying to capture, and especially nonmonotonicity, contribute an element of complexity to the subject matter. Conventions for representing information in graphs allow complex cases to be presented so that they can be better understood. In particular, complex patterns can often be understood as combinations of more simple configurations.[16] Here, we will consider three simple patterns that were important in Touretzky's work.

In the following exposition, we will confine ourselves to networks containing positive and negative IS-A links only, and will concentrate on problems of nonmonotonic inheritance. To simplify matters, we will omit Touretzky's cancellation links, and will postpone all discussion of relations until later.

The simplest pattern illustrating the nonmonotonicity of inheritance is the case of a simple exception to a general rule. In the following stock example, it is claimed that birds fly, but Tweety is listed as a non-flying bird. For obvious reasons, this pattern is called "The Tweety triangle."

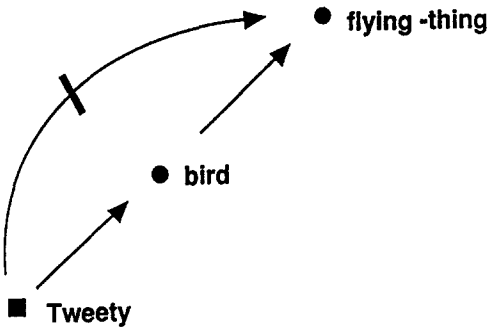

Figure 5. The Tweety triangle.

The specific information that Tweety does not fly obviously should override or *preclude* the more general information that Tweety flies. In terms of reasoning paths, this means that the positive path from **Tweety** through **bird** to **flying-thing** is overridden, while the negative path from **Tweety** to **flying-thing** represents a correct conclusion. This is the basic idea behind Touretzky's use of "inferential distance" to account for reasoning with exceptions, the idea that the structure of the graph can be used to determine when reasons should be overridden, by providing information about which reasons are more specific.

From this example, it is also clear why Fahlman decided to implement non-monotonic inheritance using a shortest-path algorithm, which also is relatively easy to implement as a parallel marker passing procedure.[17]

But Touretzky discovered examples showing that Fahlman's ideas were problematic in some respects. The following example, which shows in particular the shortest path doesn't always correspond to the best reason, is a version of what is sometimes called the *Clyde-level skip*.

Without the dotted link, this is a simple exception, like the Tweety Triangle except with more levels of classification. The dotted link represents a conclusion to which the network is committed. Since in this sense the link is redundant there may be no practical point in adding it. (On the other hand, if this link were already present in the network—for instance, if it had been the first thing that was known about Clyde—there would be no practical point in removing it, either, on learning that Clyde is a special type of elephant.) Intuitively, adding the link to the network should certainly not change the inferences that the network draws. However, this addition *will* affect the inferences that are drawn by a shortest-path algorithm, since after the addition the shortest path from **Clyde** to **gray-thing** is positive, not negative.[18] The principle is often called *cautious monotonicity*.

The sharper intuition about inheritance that emerges from examples such as this is that preclusion is always conditioned by *the most specific reason*, not by the shortest path. Whether or not

[16] This point is illustrated now by many papers in the literature. [16] is a good example.

[17] Fahlman himself doesn't put it this way; this characterization is due to Touretzky's later work.

[18] Touretzky's criticism of shortest path reasoning is closely related to a logical condition on nonmonotonic consequence relations that has been suggested by several authors, and that has sometimes been called "cautious monotonicity": the condition that if $\Gamma \mid\!\sim \phi$ and $\Gamma \mid\!\sim \psi$ then $\Gamma \cup \{\phi\} \mid\!\sim \psi$. See, for instance, [17].

the dotted link is present in Figure 6, any positive path from **Clyde** to **gray-thing** is a less specific reason for a conclusion about Clyde's color than the negative path from **Clyde** through **royal elephant** to **gray-thing**. This is because **royal elephant** is shown by the net in Figure 6 to be more specific than **elephant**.

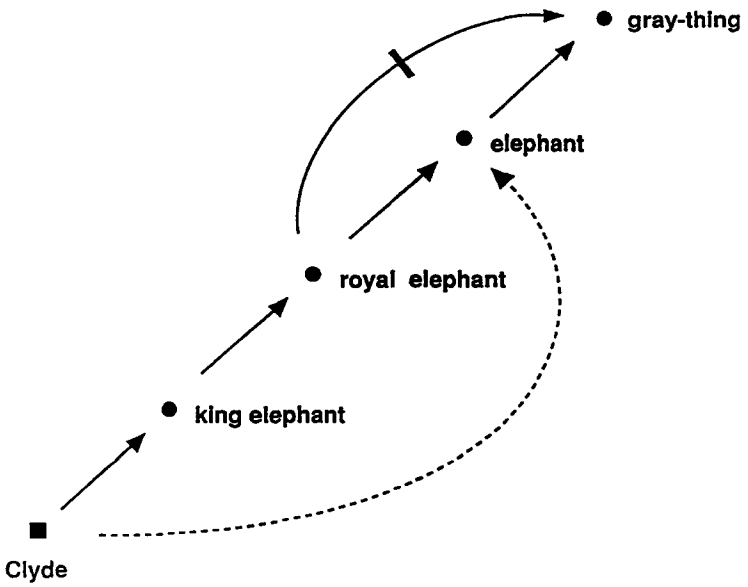

Figure 6. The Clyde-level skip.

If we allow individuals and concepts to be cross-classified, i.e., if *multiple inheritance* is permitted, and reasons can be positive and negative, then reasons can conflict. Though conflicts of this sort can perhaps be avoided in some applications, they seem to be an inescapable feature of general reasoning with defeasible generalities. Mainly through Jon Doyle's and Raymond Reiter's work (see [18] for a recent example, with references) conflicting reasons have come to be acknowledged as a major topic in nonmonotonic reasoning. The standard example of such a conflict (due to Reiter) is *The Nixon Diamond.*

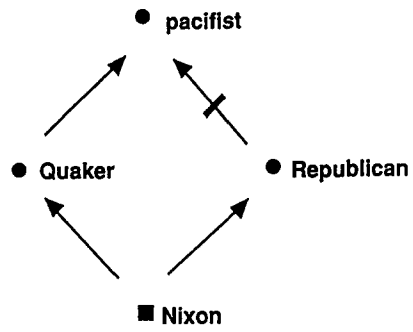

Figure 7. The Nixon diamond.

The essential difference between this example and the Tweety Triangle is that neither of the conflicting paths from **Nixon** to **pacifist** is more specific than the other.

Such conflicts, which were not noted in [6], obviously are a problem in designing inheritance algorithms. At the very least, we do not want an inheritance reasoner to infer contradictions from defeasible conflicts. At best, we would like it to distinguish conflicts like that in the Nixon Diamond, from genuine contradictions in the knowledge base, and to reason in a coherent, sound manner, making the best of available information despite conflicts.

Influenced in part by the solutions to conflicts suggested by the extensions of nonmonotonic logic and Reiter's default logic, Touretzky responds to this problem in [2] by making inheritance

relative to the choice of an "extension." An extension can be thought of as a coherent, well-motivated way of satisfying as many generalizations in the network as possible.[19] There are just two extensions of the Nixon Diamond. In both, Nixon is a Quaker and a Republican; in one he is a pacifist, and in the other he is not.

Naturally, in monotonic logic the notion of an extension is unnecessary; in reasoning from mathematical postulates there is never any need to indulge in a pattern of arbitrary guesses. Also, multiple extensions complicate the logical situation. For instance, it is not clear what to mean by the logical consequences of a premiss set Γ, or by updating Γ. Should we only talk about the logical consequences of Γ relative to an extension Φ of Γ, or should we say that A is a consequence of Γ if A is present in every extension of Γ? Should we think of update as an operation on a network, or on a network together with an extension?

Despite these intellectual complications, the need to deal with multiple extensions doesn't seem in itself to make computation hopeless. For instance, the known complexity results do not indicate that multiple extensions will in themselves introduce intractability. However, certain computational strategies that are suggested by multiple extensions can result in intractability. Touretzky placed strong constraints on his extensions having to do with *uniformity* of the choices made in constructing the extension. These led him to propose "double chaining" or "downward concatenation"[20] algorithms that calculate inheritance by seeing whether overlapping paths could be joined together. Recently, Bart Selman and Hector Levesque have shown that these algorithms will in general be intractable; for more details, see Section 4.2.2, below.

3.3. Net Notation

Before proceeding to details of Touretzky's theory, I will establish a general notation for dealing with reasoning in nets.[21] There are substantial differences between this notation and the organization of the material from [2]; it seemed better to do this than to use several different notations in presenting various network theories.

I will generally use bold italics for network items, to distinguish them, for instance, from items belonging to models or logical expressions. In particular, I will use $\boldsymbol{a,b,c}$, ... for individuals, $\boldsymbol{p,q,r}$,... for kinds, and $\boldsymbol{x,y,z}$,... for nodes in general, either individuals or kinds.

Just as a logic has a representational or syntactic part and a proof-theoretic part, an inheritance theory will have representational and inferential components. At a general level of abstraction, the representational structure of a *net* $\boldsymbol{\Gamma}$ consists of the following components.

(1) A set $\boldsymbol{I}(= \boldsymbol{I}(\boldsymbol{\Gamma}))$ of *individuals;*
(2) A disjoint set $\boldsymbol{K}(= \boldsymbol{K}(\boldsymbol{\Gamma}))$ of *kinds;*
(3) A set ***L-types*** of *link types;*
(4) A set $\boldsymbol{WF\text{-}links}(\boldsymbol{\Gamma})$ of "well-formed links;"
(5) A set $\boldsymbol{KB}(\boldsymbol{\Gamma})$ of "known links," that is, links that are part of the knowledge base of $\boldsymbol{\Gamma}$;[22]
(6) A set $\boldsymbol{A\text{-}types}(\boldsymbol{\Gamma})$ of *answer types*, or *assertion types.*

In general, the link types will vary from network to network. To have any inheritance at all there should be IS-A links. But often other sorts of links may be included: IS-NOT-A links, relational links, etc. In general, link types will not only affect algorithms, but will play a crucial part in the network's informal interpretation. Occasionally, $\boldsymbol{l}$ will stand for a link; more often, links are represented using arrows, which may or may not be decorated with type labels; $\boldsymbol{p \rightarrow q}$, for example, is a defeasible IS-A link connecting concepts $\boldsymbol{p}$ and $\boldsymbol{q}$, and $\boldsymbol{p \Rightarrow q}$ is a strict IS-A link connecting concepts $\boldsymbol{p}$ and $\boldsymbol{q}$.[23] Formally, however, we can think of a link as a triple $\langle \boldsymbol{\theta}, \boldsymbol{x,y} \rangle$, where $\boldsymbol{\theta}$ is the link type. The link $\boldsymbol{p \Rightarrow q}$, for instance, is really the triple $\langle$IS, $\boldsymbol{p,q} \rangle$.

[19] In [2], Touretzky uses the word 'expansion'; here, I use the more usual term.
[20] The two terms are interchangeable. The second is replacing the first, earlier one.
[21] The LINKUP project has produced several documents setting out general notation and definitions for inheritance theory. The version presented in the following section overlaps with that of [19]. A definitive and extensive presentation will appear in [20].
[22] Where no confusion is likely, we will simply use $\boldsymbol{\Gamma}$ to refer to $\boldsymbol{KB}(\boldsymbol{\Gamma})$.
[23] Single-shafted arrows are reserved for nonmonotonic, or defeasible links; double-shafted arrows denote strict links. The distinction between link types will be discussed below in more detail.

The set ***WF-links***(Γ) of well-formed links over Γ is obtained by imposing syntactic constraints on the set of all possible links involving elements of Γ. (For instance, it is natural to require that for $x \rightarrow y$ to be well-formed, y must be a kind.) The set ***A-types***(Γ) represents the possible answers to queries over Γ. Since units of information are stored in networks in the form of links, well-formed links represent the allowable data inputs to a network; answers, on the other hand, represent allowable data outputs. The need for a distinction between links and answer types will not be apparent from the first examples that we will give, since in these simple cases every answer type will correspond to a link type. In networks with roles, though, the distinction is needed, and similar cases can easily be imagined. For instance, there might be conjunctive answers, even though there are no conjunctive links, or, in cases where negation is implicit and closed-world reasoning is allowed, there might be negative answers, but no negative links.

The most distinctive characteristic of the network approach to reasoning is that answering a query depends on *paths* through a net. In [2], paths are identified with sequences of *signed nodes* (i.e., of nodes labeled with a member of the set $\{+,-\}$). In this exposition, we will generalize Touretzky's definition and identify paths with sequences of links.[24] As nets become more expressive, more restrictive accounts of paths need in general to be relaxed; in particular, the linearity assumption may be dropped. At the limit of this process, paths blend into logical deductions. See [21] for a case of this sort.

The inferential structure of a network Γ consists of the following additional elements.

(1) A set ***WF-paths***(Γ) of sequences whose members are drawn from ***WF-links***(Γ). These are the paths that are well-formed in Γ.

(2) An assignment to each path σ in ***WF-paths***(Γ) of an associated answer (or assertion) type $A(\sigma)$ in ***A-types***(Γ). $A(\sigma)$ is the assertion *permitted* by the path σ.

(3) A relation $\bowtie$ of inconsistency on ***A-types***(Γ).

(4) A set ***Exts***(Γ) of subsets of ***WF-paths***(Γ)—the members Φ of ***Exts***(Γ) are the sets of paths that count as inferentially correct extensions of Γ.

(5) Optionally, there may be a *support* relation $\vdash$ between ***KB***(Γ) and the assertion types of Γ. '$\Gamma \vdash A$' means that the network Γ supports the conclusion A. Support is defined using extensions: the most natural definition is that $\Gamma \vdash A$ if and only if for all $\Phi \in$ ***Exts***(Γ) there is a path $\sigma \in \Phi$ such that σ enables A. (An assertion is supported by a net if every extension of the net contains a path that permits the assertion.)

3.4. IS-A *Inheritance*

Omitting cancellation links, Touretzky's link types in [2, Chapter 2] are (defeasible) positive and negative IS-A; I will use $\rightarrow$ and $\nrightarrow$ for these link types. A link $x \rightarrow y$ is well-formed in case y is a kind. The answer types (which Touretzky does not deal with explicitly) correspond to the well-formed link types: IS(x,p) and $\overline{\text{IS}}(x,p)$ are assertions over Γ if x is a node and y a kind of Γ.

See Figure 5 for an example of the conventions I will use for diagramming networks. These differ slightly from the conventions of [2], and correspond to the most recent conventions I myself have been using. Individual nodes are square, kinds are circular. Negative IS-A links are indicated with a single heavy bar through the shaft. All nodes in Figure 5 are solid. Nodes shown in outline are used in connection with roles; see [19] for an explanation of their use.

Note that a net diagram amounts to a set ***KB***(Γ) of known links; Figure 5, for instance, corresponds to the set:

$$\{\textbf{Tweety} \rightarrow \textbf{bird}, \textbf{bird} \rightarrow \textbf{flying-thing}, \textbf{Tweety} \nrightarrow \textbf{flying-thing}\}.$$

Touretzky's IS-A nets allow well-formed paths of the form

$$\langle\langle \theta_1, x_1, y_1\rangle, \ldots, \langle \theta_n, x_n, y_n\rangle\rangle,$$

where (1) $n \geq 1$; (2) for all i, $1 \leq i < n$, $y_i = x_{i+1}$; and (3) if $\theta_i =$IS-NOT then $i = n$. In other words, paths must be simply connected, and negative links must be terminal.

[24] This generalization allows paths that are not simply connected. In Figure 6, for instance, ⟨**Clyde → king elephant, elephant → gray-thing**⟩ counts as a path in the more general sense, though not in Touretzky's.

A path is *positive* if its last link is positive, and *negative* if its last link is negative. A path $\langle\langle \theta_1, x_1, y_1\rangle, \ldots, \langle \theta_n, x_n, y_n\rangle\rangle$, is assigned the assertion-type $\text{IS}(x_1, y_n)$ if it is positive and $\overline{\text{IS}}(x_1, y_n)$ if it is negative.

Finally, two assertions $\langle \theta_1, x, p\rangle$ and $\langle \theta_2, y, q\rangle$ are contradictories in case one is an explicit denial of the other—that is, in case θ_1 and θ_2 have opposite polarity, $x = y$, and $p = q$.

3.5. Touretzky's Inheritance Definition

We now come to the inheritance definition itself. Here the "logic" of the network is specified, by characterizing the "belief-states" or extensions that correspond to a given knowledge base. Touretzky does this by two conditions; one of which requires extensions to be large enough—to be inferentially closed relative to the knowledge base—and the other of which requires them to be conservative, in not reaching conclusions that are not somehow warranted by the knowledge base.

The closure conditions, in turn, involve two key inferential relations on paths through the network: *contradiction* and *preclusion*.[25]

The conceptual work that needs to be done here is very similar to what logicians do in axiomatizing a logical language. In formulating axioms, it is often necessary, relative to the language at hand, to find syntactic characterizations of notions like contradiction. For instance, a typical principle of negative reasoning is the principle of *ex falso quod libet*:

$$[A \wedge A'] \rightarrow B,$$

that anything follows from contradictories A and A'. In formulating such a principle in a new language (perhaps one that has no explicit negation connective), it is necessary to make a decision about what pairs of formulas should count as contradictories.

In general, then, once we have decided on the expressive part of a network, we will need to clarify contradiction and preclusion before defining inheritance. Since it is usual to study nets that can express explicit negation, the former task is straightforward; preclusion is more complex.

Intuitively, a path contradicts another if it would be inconsistent to consider them both to be good arguments. For nets that have positive and negative assertion types, we can say that paths are contradictory if one of their associated assertions is the negation of the other. For reasons that will become clear below, in Section 4.1.1, we will call this *credulous* contradiction.

DEFINITION 1. CREDULOUS CONTRADICTION
σ contradicts τ relative to Φ if and only if $A(\sigma)$ and $A(\tau)$ are contradictories.

The intuition behind preclusion comes from exceptions; a path σ is precluded by a set of arguments Φ if this set provides a better reason than σ, for the opposite conclusion. Given these notions, we can build up an account of when an inference is correct relative to a set Φ of paths, which we can think of as representing the set of arguments that an agent accepts: the inference is good unless it is precluded by Φ, or is contradicted by some argument in Φ.

The formal definitions below follow [2], except in giving a definition of preclusion that is somewhat simpler, and also easier to motivate from the intuition that more specific reasons prevail. This alternative characterization of preclusion, called "off-path preclusion," was proposed in [22] as an improvement to Touretzky's definition. Off-path preclusion is more liberal than Touretzky's definition—that is, it allows more cases of preclusion. Though Sandewall's definition is simpler, Touretzky at least has not conceded that it is an improvement; see [23, pp. 480–481], for discussion.

In the following formal definitions, 'σ' and 'τ' range over the set of well-formed paths. (Including the "empty path," unless this value is explicitly excluded.) $\sigma\tau$ is the result of concatenating σ with τ. It is convenient to use a notation for paths that shows their beginning and end nodes, as well as intermediate nodes if these are important. $\sigma(x,p)$ is an arbitrary path beginning with x and ending with p; $\sigma(x,q,p)$ is an arbitrary path beginning with x, passing through q, and ending with p.

[25] The term 'preemption' has also been used for preclusion. The two terms are interchangeable.

DEFINITION 2. CONCLUSION SET
Where Φ is a set of paths, $C(\Phi)$ is the set of assertions permitted by some path in Φ.

DEFINITION 3. CREDULOUS CONTRADICTION AS A RELATION BETWEEN SETS OF PATHS AND PATHS
Φ contradicts a path σ if and only if $A(\sigma)$ contradicts some member of $C(\Phi)$.

The motivating idea of preclusion is that more arguments providing more specific reasons override those that provide less specific reasons. In the nets we are considering, arguments are paths, and the (final) reason for the conclusion of the argument will be the tail node of the corresponding paths's last link. Specificity of reasons is also determined by paths through the network, since these are the means of calculating subsumption.

Returning to Figure 5, which is the simplest case of preclusion, the argument

Tweety $\nrightarrow$ flying-thing

is better than the argument

Tweety $\rightarrow$ bird $\rightarrow$ flying-thing

because the reason of the former path, **Tweety**, is shown to be more specific than the reason of the latter, **bird**. The specificity is shown by the (one link) path from **Tweety** to **bird**.

More complicated cases of preclusion, like the following one, also need to be taken into account in constructing a general definition.

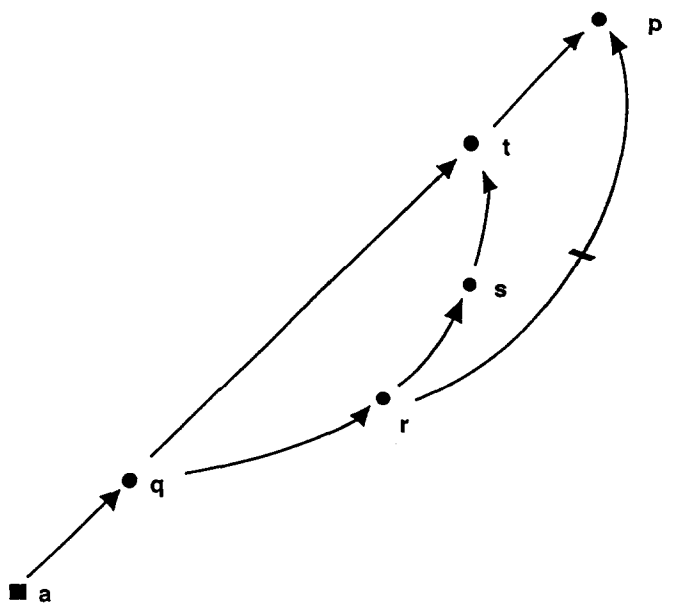

Figure 8. Preclusion.

Here, the negative path from a through q and r to p precludes the positive path from a through q and t to p because of (1) the path from a through r and s to t, which shows r to be a more specific reason about a than t, and (2) the negative link $r \nrightarrow p$, which contradicts the conclusion of the precluded path. This suggests the following definition of preclusion.

DEFINITION 4. OFF-PATH PRECLUSION AS A RELATION ON PATHS
A positive path $\sigma_1(x,q)\,\sigma_2(q,p)$ is precluded relative to Φ and Γ by a path $\tau_1(x,r)\,\langle r \nrightarrow p\rangle$ if and only if (1) $\tau_1(x,r)$ belongs to Φ, (2) $r \nrightarrow p \in \Gamma$, and (3) there is also a positive path $\tau_1(x,r)\,\tau_2(r,q)$ in Φ. A negative path $\sigma_1(x,q)\,\sigma_2(q,p)$ is precluded relative to Φ and Γ by a path $\tau_1(x,r)\,\langle r \rightarrow p\rangle$ if and only if (1) $\tau(x,r)$ belongs to Φ, (2) $r \rightarrow p \in \Gamma$, and (3) there is also a positive path $\tau_1(x,r)\,\tau_2(r,q)$ in Φ. A path is precluded in Φ if it is precluded by some path in Φ.

DEFINITION 5. OFF-PATH PRECLUSION AS A PROPERTY OF PATHS RELATIVE TO SETS OF PATHS
A path $\sigma(x,q,p)$ is precluded relative to Φ and Γ if there is a path that precludes $\sigma(x,q,p)$ relative to Φ and Γ.

With contradiction and preclusion defined, we are in a position to define the closure condition on extensions.

DEFINITION 6. COUPLED INHERITABILITY
σ is inheritable in Φ if and only if, where $\sigma = \langle l_1 \rangle \tau \langle l_2 \rangle$, (1) $\langle l_1 \rangle \tau \in \Phi$ (2) $\tau \langle l_2 \rangle \in \Phi$, and (3) Φ neither contradicts nor precludes σ. (Note: in this definition, l_1 is understood to differ from l_2; so that any inheritable path must have two links at least.)

This definition builds up larger inheritance paths by testing two paths that overlap except for their endlinks for compatibility. Touretzky chose this definition to ensure what he calls *coupling*, or coherence among the arbitrary decisions that may have to be made in constructing extensions.

Contrast the downwards strategy with an *upwards* strategy that constructs larger paths by testing the result of adding a single link to the end of a path for preclusion and inconsistency. In the network of Figure 9, the upwards strategy would allow an extension that reaches opposite conclusions about a and b in testing for inheritance of p; nothing prohibits an extension, for instance, that includes paths

$$\langle a \rightarrow s,\ s \rightarrow q,\ q \rightarrow p \rangle$$

and

$$\langle b \rightarrow s,\ s \rightarrow r,\ r \nrightarrow p \rangle.$$

This extension must also include either the path

$$\langle s \rightarrow q, q \rightarrow p \rangle$$

or the path

$$\langle s \rightarrow r, r \nrightarrow p \rangle,$$

but this second choice has no effect on paths beginning lower with a or b—the reasoning choices are "decoupled."

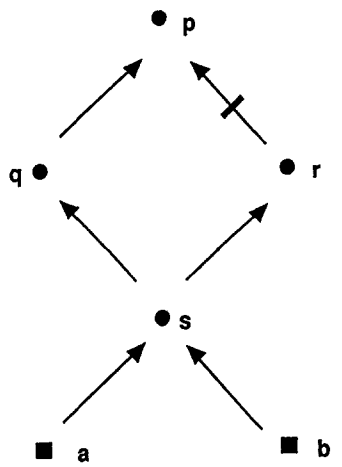

Figure 9. Coupling.

Double-chaining ensures that decisions about "general reasons," like this second choice about s's properties, will be transmitted downwards to notes that inherit from s.

We are finally in a position to define expansions.

DEFINITION 7. CREDULOUS EXPANSIONS
Where Γ is a set of links, a set of paths Φ is a (credulous) expansion of Γ if and only if (1) $\{\langle l \rangle / l \in \Gamma\} \subseteq \Phi$ and (2) Φ contains every path inheritable in Φ.

Some sets of paths that qualify as expansions of the links in a net are unwanted, because they contain gratuitous inferences. For instance, an expansion is obtained from the links in Figure 4 by adding the path $\langle$*Clyde* $\rightsquigarrow$ *gray-thing*$\rangle$ to this set. In effect, we have closed the net under inheritability by hallucinating that Clyde is an exception to the rule provided by the net.

To exclude such unwanted cases, Touretzky imposes a groundedness condition on expansions. This condition rules out unwanted paths by ensuring that every path that is actually added in the expansion must be formed by chaining subpaths using the inheritability condition.

DEFINITION 8. GROUNDED EXPANSIONS
An expansion Φ *of* Γ *is grounded in* Γ *if and only if every path in* $\Phi - \Gamma$ *is inheritable in* Φ.

DEFINITION 9. CREDULOUS EXTENSIONS
An extension of Γ *is an expansion of* Γ *that is grounded in* Γ.

With the definitions in place, Touretzky proves a number of theorems about the inheritance definition. He shows, for instance, that an extension of a net is consistent if and only if the net is consistent; that every acyclic net has at least one extension (and the proof of this theorem provides a construction of such an extension); he establishes an $O(2^N)$ upper bound for the size of an extension of an acyclic net, showing also that this bound cannot be improved; and he proves that there can be at most 3^{N^2} extensions of an acyclic net.

These proofs make a convincing case that inheritance definitions can provide a useful platform for theory development. There are parallels between the results for inheritance (at least for credulous inheritance of the sort considered in [2]) and the earlier development of a body of results about default logic. See [24] for an extended presentation of the parallels; it should also be mentioned that Etherington and Touretzky corresponded during the earlier stages of their work, and there were mutual influences.

3.6. Parallel Marker Propagation

Touretzky's formal inheritance definition provides a specification for inheritance algorithms. Much of [2] is devoted to developing an account of parallel algorithms appropriate for inheritance, including a theory of the algorithms themselves and a high level language in which inheritance procedures can be written. The central result is the correctness of certain algorithms written in this language, relative to tasks that are specified by the inheritance theory.

These results begin with a simple algorithm called UPSCAN, which calculates inheritance by computing bottom-up, beginning with an arbitrary node x, the transitive closures of positive IS-A links.[26] A marker, say TM, is placed on the nodes traversed in this process. If at any point in this process an IS-NOT-A link is encountered, the node at the head of this link is marked with FM. After UPSCAN has been applied to x, the kinds known to be true and false of x will be marked.[27]

Touretzky shows the algorithm correct for a limited class of consistent, "orthogonal" networks, in which Nixon-diamond-like conflicts are forbidden and which can be shown to have one and only one extension. In general, however, UPSCAN is incorrect.

He then explores the possibility of conditioning nonorthogonal networks to make UPSCAN work correctly, and is able to show that, by adding cancellation links to a consistent net Γ with an extension Φ a network Γ' can be constructed, such that UPSCAN will correctly compute the nodes from which an arbitrary node x inherits in Φ when run on Γ'.[28]

Touretzky formulates another PMPM algorithm, DOWNSCAN, which, given a node x, is meant to mark the nodes that inherit from x. The results about this algorithm are similar to those about UPSCAN; it is correct for a limited class of networks, and incorrect in general.

3.7. Relations in TMOIS

In other chapters of [2], Touretzky generalizes his inheritance definition and results about PMPM algorithms to networks that allow some information to be expressed about relations. The

[26] The terms 'upscan' and 'downscan' were coined by Fahlman; the first real analysis is due to Touretzky.
[27] Essentially, this is Fahlman's idea.
[28] In view of the later results of [25], however, conditioning must be intractable.

relational links he adds represent binary relations, which are interpreted generally when attached to kinds. A link $p \xrightarrow{R} q$, for instance, might stand for 'Phillips head screwdrivers fit Phillips head screws'.[29] These links can also be negated.

Universal relational statements of this sort do not involve roles or incorporate Fahlman's "virtual copy" idea. In fact, this extension seems very modest, considered against what needs to be expressed about relations in many applications. Though even this extension complicates the theoretical situation significantly, Touretzky is able to produce a plausible inheritance definition and in general is able to extend the earlier results to this case.

4. INHERITANCE THEORY

Soon after the completion of Touretzky's dissertation, a collaboration emerged in Pittsburgh between several local logicians—Charles Cross, John Horty, and Richmond Thomason—and Touretzky. This work, which was subsequently funded by the National Science Foundation, and became known as the LINKUP project,[30] has provided the background for many of the later innovations in inheritance theory. In describing developments in the field subsequent to [2], I will begin with the LINKUP work.

4.1. LINKUP

All three of the LINKUP logicians are interested in computational issues for their own sake; but I think I can speak for all of us in saying that we were attracted to the subject because of our feeling that there was a rewarding source of new ideas for logic here. Touretzky's dissertation resembled familiar logical material in many ways, but the graphlike structure of networks offered a new dimension that seemed to provide genuinely new techniques for dealing with the structure of arguments.

Philosophical logic has generally been more concerned with the creation of new logical formalisms than with developing the mathematical ramifications of known ones. Classical logic was designed to account for mathematical reasoning, and philosophical logicians have tried to extend this account to other domains, including common-sense reasoning. Thus, the goals of philosophical logic and theory in AI have much in common.[31]

In philosophical logic, motivation—articulating the intuitions behind a theory, and developing its connections to topics like reasoning and natural language—is as important as the creation of a body of theorems. A model theory and proof of completeness, for instance, does not suffice to justify a new formalism. On the one hand, there are theorems showing that a model theory can be fabricated for a very large class of logics; on the other, many logics—intuitionistic logic, the untyped lambda calculus, and the family of relevance logics, for example—were known to be interesting and important long before they had a reasonable semantic interpretation. It is motivation that shows a logical formalism to be reasonable and interesting, whether it is a proof-theoretic or a model-theoretic formalism.

From working in the theory of nonmonotonic reasoning and related areas (such as the logic of conditionals), all of us felt a need for robust, detailed intuitions that could help to motivate work in this area. We believe that network diagrams and concrete examples of inheritance reasoning provide a rich fabric of such material, which can be of just as much importance to logic as to AI.

For this reason, the logical work in the LINKUP project has concentrated on motivating inheritance definitions, proving foundational theories, exploring the space of reasonable inheritance theories, and developing extensions of the language studied in Touretzky's work. This work is closely related in methodology to the proof-theoretic tradition in logic.[32]

We have also hoped that inheritance theory would not only provide specific computationally useful ideas, but that on a larger scale it might help to address the increasing gap in the field of

[29]Touretzky uses examples like 'Elephants love zookeepers,' but in fact these don't illustrate the theory well. Whatever 'Elephants love zookeepers' means, we can't infer from it that an arbitrary elephant loves an arbitrary zookeeper, unless we have reason to think otherwise.

[30]Logic, INheritance and Knowledge UPdate.

[31]See [26] and [14].

[32]See [27], and the more recent survey in [28, Part D: "Proof Theory and Constructive Mathematics"].

knowledge representation between theory and applications, by providing an intermediate level of inquiry between the very abstract and general logical theories and actual implementations.

4.1.1. Skeptical Reasoning Strategies

Much of the conceptual complexity of Touretzky's inheritance definition arises in the management of multiple extensions, which in turn originates in the possibility of conflicting reasons. It seems that reasonable people, when faced with such conflicts, do sometimes choose one conclusion or the other; but in such cases—particularly when harboring an incorrect belief may involve some risk—it may be equally reasonable to suspend belief. Since this conservative, *skeptical* reasoning strategy will not lead to multiple extensions, it offers an alternative to Touretzky's definition that at least may have an advantage of conceptual simplicity.

Since we do not have to deal with multiple extensions, coherence is not a problem on the skeptical approach, and we can adopt a bottom-up approach to inheritance reasoning. On such an approach, the desired extension will be built up by stages (or *partial* extensions) from the original net Γ, by lengthening paths upwards, one link at a time.

A good way to see the difference between credulous and skeptical inheritance is to regard the "jump" operation that adds lengthened paths to a partial extension Φ as adding inheritance paths to Φ. If we look at things in this way, the crucial difference between skeptical inheritance and the version of credulous inheritance that we defined in Section 3.5, above is in the notion of contradiction that is used. The idea is that, by looking ahead of conflicts, a skeptic is liberal about what counts as a contradiction. For instance, in the minimal partial extension of the Nixon Diamond net presented in Figure 7, in which no paths have been added to the net, a skeptical reasoner would consider the path

$$\langle \textbf{Nixon} \rightarrow \textbf{Quaker}, \textbf{Quaker} \rightarrow \textbf{pacifist} \rangle,$$

already contradictory, because the opposed path

$$\langle \textbf{Nixon} \rightarrow \textbf{Republican}, \textbf{Republican} \nrightarrow \textbf{pacifist} \rangle,$$

is not precluded. These ideas lead to the following definition.

DEFINITION 1. SKEPTICAL CONTRADICTION AS A RELATION ON PATHS
$\sigma\langle l_1 \rangle$ is contradicted by $\tau\langle l_2 \rangle$ relative to Φ and Γ if and only if $A(\sigma)\langle l_1 \rangle$ and $A(\tau)\langle l_2 \rangle$ are contradictories, and neither $\sigma\langle l_1 \rangle$ nor $\tau\langle l_2 \rangle$ are precluded relative to Φ and Γ.

DEFINITION 2. SKEPTICAL CONTRADICTION
$\sigma\langle l_1 \rangle$ is contradicted relative to Φ and Γ if and only if $\sigma\langle l_1 \rangle$ is contradicted by some $\tau\langle l_2 \rangle$ relative to Φ and Γ.

DEFINITION 3. SKEPTICAL INHERITABILITY, OR SKEPTICAL LENGTHENING OF A PATH
$\sigma\langle l \rangle$ is inheritable in Φ and Γ (or alternatively, $\sigma\langle l \rangle$ is a skeptical lengthening of σ in Φ and Γ) if and only if (1) $\sigma \in \Phi$, (2) $l_1 \in \Gamma$, and (3) $\sigma\langle l_1 \rangle$ is neither contradicted nor precluded relative to Φ and Γ.[33]

DEFINITION 4. DEGREE
The degree of a path $\sigma(x,p)$ in a net Γ is the length of the longest sequence of links in Γ from x to p.

The notion of degree is discussed in detail in [29].

DEFINITION 5. SKEPTICAL EXTENSIONS
The (skeptical) jump of Φ is obtained by adding to Φ the skeptical lengthenings of the paths in Φ of maximal degree, relative to Φ and Γ. The skeptical extension of Γ is the union of the successive skeptical jumps obtained from Γ.

[33] It would be equivalent to omit the non-contradiction condition from (3), since if a path is precluded it is contradicted. The condition was included to emphasize the uniformity with the credulous definition.

In [29], we show that the extension of an acyclic net exists, and can be computed by a simple PMPM algorithm, which is polynomial and whose performance approaches the optimum of linearity in the depth of the net when there are few conflicted nodes. We are also able to show that inferential properties like cumulative monotony hold for the skeptical support relation, at least for singular assertions. In particular, we show that if $\Gamma \mid\!\!\sim A(l)$ then $\Gamma \cup \{l\} \mid\!\!\sim B$ if and only if $\Gamma \mid\!\!\sim B$, where l is $a \rightarrow p$ or $a \nrightarrow p$.

Since the term "skeptical inheritance" is often used in the literature to refer to the intersection of credulous extensions, it is important to notice that these two notions are not equivalent. The following "double diamond" provides a counterexample.

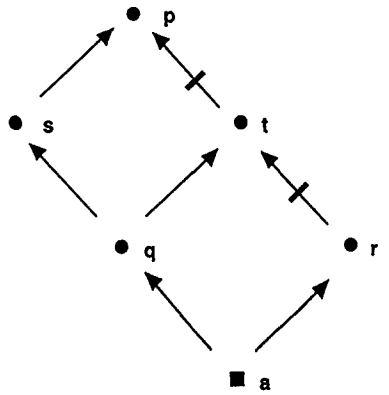

Figure 10. The double diamond.

According to skeptical inheritance, this net supports the conclusion IS(a,p). The conflicting path

$$\langle a \rightarrow q, q \rightarrow t, t \nrightarrow p \rangle$$

is not in the skeptical extension, because its initial subpath

$$\langle a \rightarrow q, q \rightarrow t \rangle$$

is contradicted.

The double diamond is a good example of how simple patterns can be combined to make more complex ones. In an interesting paper,[34] David Makinson and Karl Schlechta point out that "skeptical inheritance" is in a sense not perfectly skeptical, because it would be reasonable for a perfectly skeptical reasoner to use the path

$$\langle a \rightarrow q, q \rightarrow t, t \nrightarrow p \rangle$$

to block an argument, even though this path itself contains a subargument that is skeptically blocked. This illuminating realization of the surprisingly complex nature of skepticism is the sort of logical insight that would probably have been impossible without the methodology of inheritance theory.

4.1.2. Organizing the Theoretical Alternatives

The work that we have presented so far shows that, when exceptions and multiple inheritance are present, there are a number of alternative characterizations of inheritance. Though it is possible to compare their relative merits, it seems to us that these alternatives cannot be eliminated by showing all but one to be unreasonable on logical grounds.

Some commentators have felt that this shows the theory of nonmonotonic inheritance to be incoherent, at least as a logical program. Probably the title of one of our papers, "A clash of intuitions," helped to create this impression. Our own feeling is that even the monotonic tradition

[34] See [30].

in logic has profited from developing alternative intuitions systematically and comparing them—the difference between classical and intuitionistic logic is a case in point. We should not be surprised to find more of the same thing in nonmonotonic reasoning.

The important thing, when faced with such alternatives—especially when the number of alternatives becomes large, as in the case of modal logic—is to be able to manage the explosion of cases intelligently, so that the similarities and differences are clear. One of the major goals of [23] was to begin this process of case management. Since then, we have carried the process further (the shared definitions in the presentation of credulous and skeptical versions of inheritance in the present paper is an example), but have not yet prepared an extended, definitive presentation. An extended version of [23] is in preparation. [20] also develops the ideas.

4.1.3. Monotonic Inheritance

In Section 4.2.1, below, I try to explain why, though, of course, a general model theoretic semantics for inheritance systems is very desirable, developing such a semantics is a large-scale research problem that we have chosen to postpone until we understand the proof theoretic aspects better.

One special case, though, is not problematic from a logical point of view; this is the case of monotonic inheritance. Though this case may be very simple, working out the details would at least provide a pattern for how to interpret an inheritance network in a logic.

To our surprise, as the theory emerged, monotonic inheritance turned out to be not at all trivial. Even in the simplest cases a nonclassical logic is required, and when roles and relations are present the inheritance definition is complex, both conceptually and computationally. In this section I will discuss only the case where IS-A and IS-NOT-A links are present.

Here the inheritance definition is in fact simple; it involves taking the transitive upwards closure of positive IS-A links, and—for negative paths—traversing as well exactly one negative link, and an arbitrary number of *inverse* IS-A links. The following picture shows a negative path of type $\overline{\text{IS}}(a,p)$; strict links, as usual, are shown with double shafted arrows.

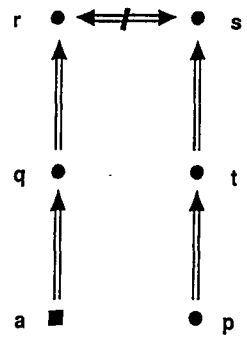

Figure 11. A strict negative path.

The nonclassical nature of these nets is illustrated by networks with contradictions, such as the following example.

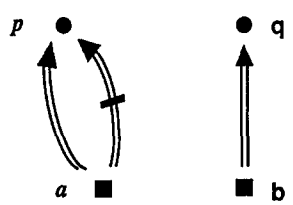

Figure 12. Need for 4-valued logic.

Because of the inconsistent information entered in the links from a, this net supports both $\text{IS}(a,p)$ and its negation $\overline{\text{IS}}(a,p)$. However, there is no path in the network at all from a to q, so there is no path permitting either $\text{IS}(a,q)$ or $\overline{\text{IS}}(a,q)$.

Thus, in order to obtain an interpretation that is sound and complete for this inheritance definition, we need to use a non-classical logic. In [31], we show that a four-valued logic is appropriate;[35] there are also nice relations between the inheritance definition and a proof-theoretic account using Gentzen-style rules.

4.1.4. Mixing Strict and Defeasible Inheritance

In [33], Brachman points out that a representation system cannot accommodate defined concepts if the only connections between concepts that it can express are defeasible. In cases where the definition provides a criterion for applying the defined concept, we don't want to leave room for exceptions—for instance, when we define WIFE as MARRIED WOMAN, we don't want to leave open the possibility that there are exceptional wives that aren't married. Some readers have taken Brachman's paper to be a critique of nonmonotonicity; but its challenge to nonmonotonic inheritance is to produce a coherent combination of strict with defeasible links. As we have seen, interactions between primitives can cause difficulties; so our chief concern in developing such a mixed system will be interactions between strict and defeasible links.

In [34], an inheritance definition is presented that mixes strict inheritance with the skeptical system of [29]. The crucial idea in defining mixed inheritance is to build "strict look-ahead" into the definition of contradiction; we need to say that paths are skeptically contradictory in case any of their strict extensions are contradictory. The details of the definition are provided in [34].

In defining mixed systems of this kind, it provides a useful check to show that the system specializes to the components that it combines. For instance, we would want to show that the special case of mixed inheritance for nets all of whose links are strict is the system of [31], and the case for nets all of whose links are defeasible is the system of [29]. These results can be established for the system of [34].

In more recent work, we have investigated the mixed algorithm. Though it is certainly polynomial, it appears that the mixed case can place more computational strain on a PMPM than the pure defeasible case.[36] If this can be backed up with a solid complexity result, it could provide some force to the interpretation of Brachman's paper as an argument against nonmonotonicity.

Despite the threat of intractability, we hope to extend the mixed systems to obtain a more expressive inheritance definition that allows the capability of making a healthy spectrum of definitions. The proven usefulness of definitional capability in knowledge representation, through the extensive application of KL-ONE like systems, provides good evidence for incorporating definitions in a representation system.[37] In particular, when definitions are present an inheritance algorithm becomes a *classifier* which is able to perform more complex tasks, such as recognition.

Extending the work on mixed inheritance, [21] considers extensions of inheritance that contain conjunctive and negative nodes. Though inheritance in such a strong system is intractable, this work at least provides a general framework for boolean definitions. We have not yet worked out a theory of quantificational definitions involving roles, of the sort considered in KL-ONE.[38]

4.2. Some Trends

This section will try to provide some perspective on selected trends in the current work on inheritance theory, and to correct any impression the previous exposition may have given that all work on inheritance is due to associates of Touretzky. Work in this area is widespread and diverse—so much so that in the space that is available here I cannot try to give an adequate survey. Even so, I want to apologize for omissions in the coverage provided in this section; these are due more to incompleteness in my own knowledge than anything else. I will try to find an opportunity to correct these omissions on a later occasion.

---

[35] This four-valued logic had independently been proposed for modeling knowledge bases; see [32]. It has in fact occurred to a number of people that localizing the harm done by inconsistencies might be useful in knowledge representation.

[36] See [35].

[37] In retrospect, one of the chief shortcomings of NETL is its inability to support definitions.

[38] Definitions like 'A full hotel is a hotel, all of whose rooms are occupied.'

4.2.1. Relations to Logic and Probability

I tried to illustrate ways in which inheritance diagrams provide a subtle and fine-grained tool for creating and examining questions about reasoning. Tarskian model theory, on the other hand, represents a relatively coarse-grained approach, due to the requirement that the truth-conditions of complex logical expressions have to depend uniformly somehow on the truth-conditions of their parts. This coarse-grainedness persists even when the classical models are extended to include possible worlds, higher-order domains, nonclassical truth values, and the like. Of course, this is also a great advantage of the model theoretic approach, since in obliterating distinctions it secures greater focus and power. Just because classical model theory is so different from proof theory, the completeness result is surprising, impressive, and useful.

Even compared with classical proofs, networks provide additional structure that is relevant to reasoning; Touretzky's inferential distance principle is a good example of this. Because of this, we have to be prepared for complications in relating inheritance networks to logics. We have seen one example of this in even the simplest monotonic case, discussed above in Section 4.1.3. This shows, I believe, that in interpreting inheritance networks we have to be cautious in considering relatively simple interpretation schemes like that of [37]; this interpretation, to be sure, is sound, but it validates an inference that does not seem correct from a path-based perspective. This argument may also show that nonmonotonic logics based on four-valued logic would be appropriate frameworks for interpreting nonmonotonic inheritance networks.[39]

By now there is a diverse literature concerning the logical interpretation of nonmonotonic inheritance networks. Perhaps the earliest systematic interpretations were due to Etherington; the results are brought together in [24, Chapter 4]. Etherington presents a translation of a system of nonmonotonic logic into Reiter's default logic, and establishes a well-behaved correspondence between the extensions of an acyclic network and extensions of the corresponding default theory.

Results of this kind are very encouraging; without such connections to logical theories of defeasible reasoning, the claim that inheritance theories provide a bridge between logical theories and applications would be unsupported. But several issues arise in evaluating Etherington's interpretation, and in fact apply generally to interpretations of networks.

Etherington's translation into default logic is not *modular;* the translation of a link has to explicitly encode all the ways in which the rule encoded in the link could be precluded; thus, this translation does not depend on the terms in the link itself, but on its context in the net. The translation may change if the net is updated elsewhere.

Etherington has a reply to this objection of Touretzky's to his interpretation, and the computational issue of whether modularity is undesirable is debatable. I believe, though, that non-modular translations are unlikely to provide *logical* illumination of the principle that more specific reasons should dominate less specific ones. This does have the marks of a logical principle, and it would be much better to make it a fairly deep consequence of the model theory than to build it into the interpretation in an *ad hoc* manner. If this is correct, the existing logical theories of defeasible reasoning would need to be modified somehow in order to provide a really illuminating framework for inheritance. My own preference would be to use this as a guide to research in constructing the logical theories—but this may be due partly to the fact that, as a logician, I tend to look for opportunities to construct new logics.

An example of such an approach, which introduces partial models and three-valued logic into the interpretations, is developed by Sandewall in [39], and applied to inheritance networks by Doherty in [40]. The extra model-theoretic structure that is provided by partiality does seem to provide a way of building specificity into the model theories (roughly, a rule is more specific if it applies in a wider class of partial models), and in fact the Sandewall-Doherty approach does seem to provide a more principled explanation of some cases, at least, of preclusion.[40]

[41] is an extended discussion of inheritance networks and their semantics, which develops a number of promising ideas. In particular, there is an extended treatment of some special cases that lend themselves to semantic interpretation; Krishnaprasad shows, for instance, that tree-

[39] Some of the points made in this paragraph are expanded in [38].

[40] In unpublished work, Nicholas Asher and Michael Morreau have also developed a semantics for defeasible reasoning that involves partial information, and that may provide a useful vehicle for net interpretation.

structured class-property networks can be treated naturally using circumscription theory. The circumscriptive interpretation is also discussed in [42].

Another approach to the use of circumscription in interpretating networks is developed in [43]; Haugh's goal to modify one of the more restrictive versions of circumscription for this task.

The autoepistemic interpretation of inheritance networks has been developed by Michael Gelfond and Halina Przymusinska; see [44]. This work is perhaps the most logically advanced of the interpretations of inheritance theory in a known nonmonotonic formalism. There are some interesting positive theorems about the adequacy of the translations, and some useful discussion of the role that the interpretation can play in explaining patterns of inheritance reasoning.

Eric Gregoire has investigated interpretations of networks in hierarchical and stratified logic programs; see [45] and [46]. Brewka's work in [47] provides a circumscriptive interpretation.

Though to some extent, the work in nonmonotonic inhertance seems to be developing intuitions about defaults that have more to do with conventional stereotypes than with probability, interpretations of nonmonotonic reasoning based on probability theory can be tested on inhertance networks. Judea Pearl has developed this idea in [48] and [49]; the goal of this reasearch is to show that the techniques he has developed in uncertainty management will yield illumination and computationally valuable insights relatingto inheritance. In fact, the work I have seen certainly does deliver illumination. Though to produce plausible inheritance reasoning it seems that complicated probabilistic models involving "maximal entropy" assumptions are needed, these models also validate patterns of reasoning involving irrelevant conditions that standard inheritance reasoners would not be able to capture—the conclusion that red birds fly, for instance, given just the information that birds fly.

Hector Geffner's recent work shows that ideas from probabalistic approaches to semantics and knowledge representation can also be powerful and enlightening tools in accounting for default reasoning in general, and inheritance networks in particular. In [50], a general semantic theory is developed that is similar in some respects to the semantics proposed by philosophical logicians for "subjunctive conditionals." The application of these ideas to inheritance is discussed briefly, and further developed in another publication, [51]. One nice feature of Geffner's approach is the importance of causal reasoning in his accounts; despite the importance of causality in reasoning applications, other work in inheritance theory has not had much to say about this topic.

4.2.2. Complexity

As in other areas of knowledge representation the use of complexity techniques in inheritance theory has grown more sophisticated, and areas of intractability have been discovered.

I have already mentioned Fahlman's informal argument that in optimal cases the parallel algorithms he proposed would perform in time linear to the depth of the knowledge base. Touretzky's dissertation [2] formalized the reasoning tasks, and showed that Fahlman's argument worked in special cases, but was problematic for networks with many multiple extensions. Touretzky proposed a process of conditioning the network as a solution. In the later work of [29] and [52], Horty and Touretzky established the tractability of the upwards, skeptical inheritance definition.

In [53], Lynn Stein presents an ingenious polynomial algorithm for computing the intersection of extensions in an upward, credulous inheritance system, with on-path preclusion.

Soon afterwards, complexity specialists began to work on inheritance problems. Bart Selman and Hector Levesque established, by a reduction to an NP-complete graph-theoretic problem,[41] that the problem of finding some extension of an acyclic net is NP-complete for downwards, credulous inheritance reasoning. Other results that they claim in this paper indicate that downwards inheritance reasoning will in general be intractable, while upwards inheritance reasoning will be tractable.[42]

As the space of inheritance systems is mapped out more accurately, we believe that tractable versions of more expressive inheritance systems will be discovered, but that the complexity of these systems may increase as expressive power is added. Thus, for instance, [56] reports an

[41] The problem of "forbidden pairs;" see [54].

[42] Since I have not seen [55], the longer work in which the detailed proofs are presented, I am being cautious and a bit vague in reporting the general results.

$O(l^5n^{10})$ upper bound for inheritance in a monotonic system with relational reasoning, where l is the number of relational link types and n is the number of nodes. This bound may well be loose, but even with better bounds we expect to discover natural inheritance problems that are nonlinear. Even for systems that are theoretically tractable, these results indicate that inheritance systems would have to be tested in practice to establish their performance qualities, especially where very large knowledge bases are a concern.

4.2.3. Roles and Relations

Fahlman felt that one of the most important, unifying insights of his work was his account of role inheritance as "virtual copying." Touretzky's account of relations in [2] omitted any theory of roles or virtual copying. In view of the importance of roles in representations, this omission needs to be corrected.

In our[43] own work, we decided to approach the problem in two stages: first by developing a theory of *monotonic* inheritance for the relevant constructs, and then constructing the analogous theory of nonmonotonic inheritance. Even the simplest cases of nonmonotonic inheritance have turned out to be conceptually hard. So it seems good to adopt a methodology of clarifying the nonmonotonic theory of a phenomenon before attempting the harder case. We believe that our experience with roles and relations has justified this work plan, especially since even the monotonic theory of roles is subtle.

Our desire for a theory of roles that will support a uniform logical interpretation has led us to treat them somewhat differently from Fahlman. Restricting ourselves to single-valued roles, we think of roles as expressing partial functions from individuals to individuals, and treat role values of kinds as *dependent* kinds. The information in links attached to these dependent kinds is interpreted using a logical formula referring to the non-role-value "ancestor" from which a role is obtained.[44] Figure 13, for instance, contains the logical information in the table of formulas following it.

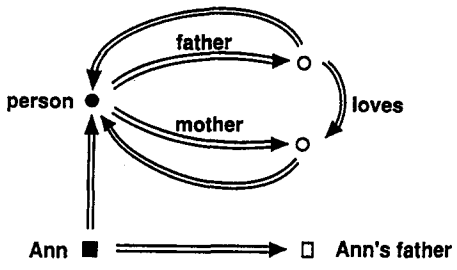

Figure 13. A net with roles.

1. $(\forall x)\,[P(x) \rightarrow (\exists y)[y = f(x)]]$
2. $(\forall x)\,[P(x) \rightarrow (\exists y)[y = m(x)]]$
3. $(\forall x)\,[P(x) \rightarrow P(f(x))]$
4. $(\forall x)\,[P(x) \rightarrow P(m(x))]$
5. $(\forall x)\,[P(x) \rightarrow L(f(x), m(x))]$
6. $P(a)$
7. $\exists x[x = f(a)]$

To formalize the idea of "virtual copying" we have introduced complex assertion types with sequences of roles as arguments. For instance, the statement IS(**mf,a,f,p**) might correspond to a statement like 'Ann's mother's father is a person's father.' We would like the net in Figure 13 to support this conclusion. Using roles as arguments in statements, which are then verified by inheritance algorithms that need not introduce new structures into the net, is a way of implementing the virtual copy idea. However, a little thought about the way these paths must look makes it clear that a path must somehow contain information about the roles that it has traversed. And

[43] Here, "we" are Thomason and Touretzky.

[44] We require that there will be a unique such ancestor. Role value nodes are shown in outline; non-role-value, or independent nodes, are solid.

to implement this idea on a PMPM we would need, in effect, we need to propagate stacks of roles rather than marker bits. Unfortunately, this complicates Fahlman's idea that the depth of a net can be calculated by looking simply at chains of IS-A links.

The details of this idea are described in [19]. Complexity results in the low polynomial range are presented in [57], for some special cases of the inheritance problem. The complexity of the general case, where identity links can connect roles attached to kinds, has not yet been determined.

As far as I know, there is no good reference yet to the inheritance theory of nonmonotonic roles. Since the monotonic case is still being worked out, the LINKUP group, at least, has not written anything about the nonmonotonic case. But we hope to address this problem very soon.

REFERENCES

1. R. Brachman, On the epistemological status of semantic networks, In *Associative Networks: Representation and Use of Knowledge by Computers,* (Edited by N. Findler), pp. 3–50, Academic Press, New York, (1979).
2. D. Touretzky, *The Mathematics of Inheritance Systems,* Morgan Kaufmann, San Mateo, CA, (1986).
3. G. Steele, Jr., *Common* LISP: *The Language,* 2nd edn., Digital Press, Bradford, MA, (1990).
4. D. Flickinger, Lexical rules in the hierarchical lexicon, Ph.D. Dissertation, Linguistics Department, Stanford University, Palo Alto, CA, (1987).
5. R. Evans and G. Gazdar, The DATR papers: February 1990, Cognitive Science Research Paper CSRP 139, School of Cognitive and Computing Science, University of Sussex, Brighton, England, (1990).
6. S. Fahlman, NETL: *A System for Representing and Using Real-World Knowledge,* MIT Press, Cambridge, MA, (1979).
7. W.D. Hillis, *The Connection Machine,* MIT Press, Cambridge, MA, (1985).
8. M. Evett, J. Hendler and L. Spector, Parallel knowledge representation on the connection machine, Technical Report CS–TR–2409, Computer Science Department, University of Maryland, (1990).
9. L. Spector, J. Hendler, and M. Evett, Knowledge representation in PARKA, Technical Report CS–TR–2410, Computer Science Department, University of Maryland, (1990).
10. J. Doyle and R. Patil, Two dogmas of Knowledge Representation: Language restrictions, taxonomic classifications, and the utility of representation services, MIT/LCS Technical Report, (1989).
11. P. Winston, *Artificial Intelligence,* 2nd edn., Addison-Wesley, Reading, MA, (1984).
12. M. Minsky, A framework for representing knowledge, In *Mind Design,* (Edited by J. Haugeland), pp. 95–28, MIT Press, Cambridge, MA, (1981).
13. S. Shieber, *An Introduction to Unification-Based Approaches to Grammar,* Center for the Study of Language and Information, Palo Alto, CA, (1986).
14. J. McCarthy, Artificial intelligence, logic and formalizing common sense, In *Philosophical Logic and Artificial Intelligence,* (Edited by R. Thomason), pp. 161–183, Kluwer, (1990).
15. M. Ginsberg, Ed., *Readings in Nonmonotonic Reasoning,* Morgan Kaufmann, San Mateo, CA, (1987).
16. R. Loui, Defeat among the arguments: A system of defeasible inference, *Computational Intelligence* **3**, 100–106 (1987).
17. S. Kraus, D. Lehmann and M. Magidor, Nonmonotonic reasoning, preferential models and cumulative logics, *Artificial intelligence* **44**, 167–207 (1990).
18. J. Doyle and M. Wellman, Impediments to universal preference-based default theories, In *KR'89: Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning,* pp. 84–102, Morgan Kaufmann, San Mateo, (1989).
19. R. Thomason and D. Touretzky, Inheritance theory and networks with roles, In *Principles of Semantic Networks,* (Edited by John Sowa), Morgan Kaufmann, (1990, forthcoming).
20. J. Horty, Some direct theories of nonmonotonic inheritance, In *Handbook of Logic in Artificial Intelligence and Logic Programming,* (Edited by D. Gabbay *et al.*) (to appear).
21. J. Horty and R. Thomason, Boolean extensions of inheritance networks, In *AAAI-90: Proceedings of the Ninth National Conference on Artificial Intelligence,* pp. 633–639, AAAI Press & MIT Press, (1990).
22. E. Sandewall, Non-monotonic inference rules for multiple inheritance with exceptions, In *Proceedings of the IEEE 74,* pp. 1345–1353, (1986).
23. D. Touretzky, J. Horty and R. Thomason, A clash of intuitions: The current state of nonmonotonic multiple inheritance systems, In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence,* (Edited by J. McDermott), pp. 476–482, Morgan Kaufmann, Los Altos, (1987).
24. D. Etherington, *Reasoning with Incomplete Information,* Morgan Kaufmann, San Mateo, CA, (1988).
25. B. Selman and H. Levesque, The tractability of path-based inheritance, Technical Report, Department of Computer Science, University of Toronto, Toronto, Ontario, (1989).
26. J. McCarthy, Programs with common sense, In *Proceedings of the Teddington Conference on the Mechanization of Thought Processes,* Her Majesty's Stationary Office, London, (1959).
27. H. Curry, *Foundations of Mathematical Logic,* McGraw-Hill, New York, (1963).
28. J. Barwise, Editor, *Handbook of Mathematical Logic,* North-Holland, Amsterdam, (1977).
29. J. Horty, R. Thomason, and D. Touretzky, A skeptical theory of inheritance in nonmonotonic semantic nets, *Artificial Intelligence* **42**, 311–348 (1990).

30. D. Makinson and K. Schlechta, Floating conclusions and zombie paths, *Artificial Intelligence* (to appear).
31. R. Thomason, J. Horty, and D. Touretzky, A calculus for inheritance in monotonic semantic nets, Technical Report No. CMU-CS-86-138, Department of Computer Science, Carnegie Mellon University, (1986).
32. N. Belnap, How a computer should think, In *Contemporary Aspects of Philosophy*, (Edited by G. Ryle), pp. 30–56, Oriel Press, Oxford, (1977).
33. R. Brachman, "I lied about the trees" or, defaults and definitions in knowledge representation, *The AI Magazine* 6, 80–93 (1985).
34. J. Horty and R. Thomason, Mixing strict and defeasible inheritance, In *AAAI-88: Proceedings of the Seventh National Conference on Artificial Intelligence*, Vol. 2, pp. 427–432, Morgan Kaufmann, San Mateo, CA, (1988).
35. D. Touretzky and R. Thomason, An inference algorithm for networks that mix strict and defeasible inheritance, In *Methodologies for Intelligent Systems, Proceedings of the Fifth International Symposium on Methodologies for Intelligent Systems*, (Edited by Z. Ras *et al.*), pp. 212–225, North-Holland, Amsterdam, (1990).
36. D. Touretzky and R. Thomason, Nonmonotonic inheritance and generic reflexives, In *AAAI-88 Proceedings of the Seventh National Conference on Artificial Intelligence*, Vol. 2, pp. 433–438, Morgan Kaufmann, San Mateo, CA, (1988).
37. P. Hayes, The logic of frames, In *Frame Conceptions and Text Understanding, (Edited by D. Metzing)*, pp. 46–61, Walter de Gruyter & Co., Berlin & New York, (1979).
38. R. Thomason and J. Horty, Logics for nonmonotonic inheritance, In *Proceedings of the Second International Workshop on Non-Monotonic Reasoning, Grassau (FRG), June 13–15, 1988)*, (Edited by M. Reinfrank, J. de Kleer, M. Ginsberg, and E. Sandewall), subseries LNAI, pp. 220–237, Springer-Verlag LNCS, (1988).
39. E. Sandewall, The semantics of non-monotonic entailment defined using partial interpretations, In *Proceedings of the Second International Workshop on Non-Monotonic Reasoning, Grassau (FRG) (June 13–15, 1988)*, (Edited by M. Reinfrank, J. de Kleer, M. Ginsberg, and E. Sandewall), subseries LNAI, pp. 27–41, Springer-Verlag LNCS, (1988).
40. P. Doherty, A correspondence between inheritance properties and a logic of preferential entailment, In *Methodologies for Intelligent Systems, Proceedings of the Fourth International Symposium on Methodologies for Intelligent Systems*, (Edited by Z. Ras *et al.*), pp. 395–402, North-Holland, Amsterdam, (1989).
41. T. Krishnaprasad, The semantics of inheritance networks, Ph.D. Dissertation, Computer Science Department, State University of New York at Stony Brook, Stony Brook, (1989).
42. T. Krishnaprasad, M. Kifer, and D. Warren, On the circumscriptive semantics of inheritance networks, In *Methodologies for Intelligent Systems, Proceedings of the Fourth International Symposium on Methodologies for Intelligent Systems*, (Edited by Z. Ras *et al.*), pp. 448–456, North-Holland, Amsterdam, (1989).
43. B. Haugh, Tractable theories of multiple defeasible inheritance in ordinary non-monotonic logics, In *AAAI-88 Proceedings of the Seventh National Conference on Artificial Intelligence*, Vol. 2, pp. 421–426, Morgan Kaufmann, San Mateo, CA, (1988).
44. M. Gelfond and H. Przymusinska, Inheritance reasoning in autoepistemic logic, In *Methodologies for Intelligent Systems, Proceedings of the Fourth International Symposium on Methodologies for Intelligent Systems*, (Edited by Z. Ras *et al.*), pp. 419–429, North-Holland, Amsterdam, (1989).
45. E. Gregoire, Skeptical theories of inheritance and nonmonotonic logics, In *Methodologies for Intelligent Systems, Proceedings of the Fourth International Symposium on Methodologies for Intelligent Systems*, (Edited by Z. Ras *et al.*), pp. 430–438, North-Holland, Amsterdam, (1989).
46. E. Gregoire, About the logical interpretation of ambiguous inheritance hierarchies, In Technical Report CS–TR–2452, Computer Science Department, University of Maryland, (1990), also in *Proceedings of the Third International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems* (to appear).
47. G. Brewska, The logic of inheritance in frame systems, In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, (Edited by J. McDermott), pp. 483–488, Morgan Kaufmann, Los Altos, (1987).
48. J. Pearl, Deciding consistency in inheritance networks, Technical Report 870053 (R-96), UCLA Cognitive Systems Laboratory, (1987).
49. J. Pearl, Probabilistic semantics for nonmonotonic reasoning: A survey, Presented at the *KR'89: Proceedings of the First International Conference on Principles of Knowledge Representation and Reasoning*, (Edited by R. Brachman, *et al.*), pp. 505–516, Morgan Kaufmann, San Mateo, (1989).
50. H. Geffner, Default reasoning: Causal and conditional theories, Ph.D. Dissertation, Computer Science Department, UCLA, Los Angeles, (1989); Also Technical Report 137, Cognitive Systems Laboratory, UCLA, Los Angeles, (1989).
51. H. Geffner and T. Verna, Inheritance = chaining + defeat, In *Methodologies for Intelligent Systems, Proceedings of the Fourth International Symposium on Methodologies for Intelligent Systems*, (Edited by Z. Ras *et al.*), pp. 411–418, North-Holland, Amsterdam, (1989).
52. J. Horty, R. Thomason and D. Touretzky, A skeptical theory of inheritance in nonmonotonic semantic nets, In *AAAI-87: Proceedings of the Sixth National Conference on Artificial Intelligence*, Vol. 2, pp. 358–363, Morgan Kaufmann, San Mateo, CA, (1987).

53. L. Stein, Skeptical inheritance: Computing the intersection of credulous extensions, In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence,* (Edited by N. Srindharan), pp. 1153–1158, Morgan Kaufmann, Los Altos, (1989).
54. H. Gabow, S. Maheshwari and L. Osterweil, On two problems in the generation of program test paths, In *IEEE Transactions on Software Engineering,* pp. 227–231, (1976).
55. B. Selman and H. Levesque, The tractability of path-based inheritance, In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence,* (Edited by N. Srindharan), pp. 1140–1145, Morgan Kaufmann, Los Altos, (1989).
56. R. Thomason, Completeness proofs for monotonic nets with relations and identity, In *Methodologies for Intelligent Systems, Proceedings of the Fourth International Symposium on Methodologies for Intelligent Systems,* (Edited by Z. Ras *et al.*), pp. 523–532, North-Holland, Amsterdam, (1989).
57. R.A. de T. Guerreiro, A. Hemerly, and Y. Shoham, On the complexity of monotonic inheritance with roles, In *AAAI-90 Proceedings of the Ninth National Conference on Artificial Intelligence,* pp. 627–632, AAAI Press & MIT Press, Menlo Park, CA & Cambridge, MA, (1990).
58. R. Thomason, J. Horty, and D. Touretzky, A calculus for inheritance in monotonic semantic nets, In *Methodologies for Intelligent Systems, Proceedings of the Second International Symposium on Methodologies for Intelligent Systems,* (Edited by Z. Ras and M. Zemankova), pp. 280–287, North-Holland, Amsterdam, (1987).
59. D. Touretzky, J. Horty, and R. Thomason, A clash of intuitions: The current state of nonmonotonic multiple inheritance systems (abbreviated version), In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence,* (Edited by J. McDermott), pp. 476–482, Morgan Kaufmann, Los Altos, (1987).
60. D. Touretzky, J. Horty and R. Thomason, Issues in the design of nonmonotonic multiple inheritance systems, Technical Report, Computer Science Department, Carnegie Mellon University (to appear).